fc	fractional capability variable
x, g, a, b	expression variable
k	integer variable
el	array-element variable
l	location variable
M	matrix variable

m	$::=$			matrix expressions
	$\|$	M		matrix variables
	$\|$	$m + m'$		matrix addition
	$\|$	$m\, m'$		matrix multiplication
	$\|$	(m)	S	
f	$::=$			fractional capability
	$\|$	fc		variable
	$\|$	1		whole capability
	$\|$	$\frac{1}{2} \cdot f$		
t	$::=$			linear type
	$\|$	**unit**		unit
	$\|$	**bool**		boolean (true/false)
	$\|$	**int**		63-bit integers
	$\|$	**elt**		array element
	$\|$	f **arr**		arrays
	$\|$	f **mat**		matrices
	$\|$	$!t$		multiple-use type
	$\|$	$\forall\, fc.t$	bind fc in t	frac. cap. generalisation
	$\|$	$t \otimes t'$		pair
	$\|$	$t \multimap t'$		linear function
	$\|$	(t)	S	parentheses
p	$::=$			primitive
	$\|$	**not**		boolean negation
	$\|$	$(+)$		integer addition
	$\|$	$(-)$		integer subtraction
	$\|$	$(*)$		integer multiplication
	$\|$	$(/)$		integer division
	$\|$	$(=)$		integer equality
	$\|$	$(\langle)$		integer less-than
	$\|$	$(+.)$		element addition
	$\|$	$(-.)$		element subtraction
	$\|$	$(*.)$		element multiplication
	$\|$	$(/.)$		element division
	$\|$	$(= .)$		element equality
	$\|$	$(< .)$		element less-than
	$\|$	**set**		array index assignment
	$\|$	**get**		array indexing
	$\|$	**share**		share array
	$\|$	**unshare**		unshare array
	$\|$	**free**		free arrary
	$\|$	**array**		Owl: make array
	$\|$	**copy**		Owl: copy array
	$\|$	**sin**		Owl: map sine over array
	$\|$	**hypot**		Owl: $x_i := \sqrt{x_i^2 + y_i^2}$

	$\mid$	**asum**		BLAS: $\sum_i \lvert x_i \rvert$
	$\mid$	**axpy**		BLAS: $x := \alpha x + y$
	$\mid$	**dot**		BLAS: $x \cdot y$
	$\mid$	**rotmg**		BLAS: see its docs
	$\mid$	**scal**		BLAS: $x := \alpha x$
	$\mid$	**amax**		BLAS: $\mathrm{argmax}\, i :\ x_i$
	$\mid$	**setM**		matrix index assignment
	$\mid$	**getM**		matrix indexing
	$\mid$	**shareM**		share matrix
	$\mid$	**unshareM**		unshare matrix
	$\mid$	**freeM**		free matrix
	$\mid$	**matrix**		Owl: make matrix
	$\mid$	**copyM**		Owl: copy matrix
	$\mid$	**copyM_to**		Owl: copy matrix onto another
	$\mid$	**sizeM**		dimension of matrix
	$\mid$	**trnsp**		transpose matrix
	$\mid$	**gemm**		BLAS: $C := \alpha A^{T?} B^{T?} + \beta C$
	$\mid$	**symm**		BLAS: $C := \alpha AB + \beta C$
	$\mid$	**posv**		BLAS: Cholesky decomp. and solve
	$\mid$	**potrs**		BLAS: solve with given Cholesky
v	$::=$			values
	$\mid$	p		primitives
	$\mid$	x		variable
	$\mid$	$()$		unit introduction
	$\mid$	**true**		true
	$\mid$	**false**		false
	$\mid$	k		integer
	$\mid$	l		heap location
	$\mid$	el		array element
	$\mid$	**Many** v		!-introduction
	$\mid$	**fun** $fc \to v$		frac. cap. abstraction
	$\mid$	$v[f]$		frac. cap. specialisation
	$\mid$	(v, v')		pair introduction
	$\mid$	**fun** $x : t \to e$	bind x in e	abstraction
	$\mid$	**fix** $(g, x : t, e : t')$	bind $g \cup x$ in e	fixpoint
	$\mid$	(v)	S	parentheses
e	$::=$			expression
	$\mid$	p		primitives
	$\mid$	x		variable
	$\mid$	**let** $x = e$ **in** e'	bind x in e'	let binding
	$\mid$	$()$		unit introduction
	$\mid$	**let** $() = e$ **in** e'		unit elimination
	$\mid$	**true**		true
	$\mid$	**false**		false

	|	$\textbf{if}\, e\, \textbf{then}\, e_1\, \textbf{else}\, e_2$		if
	|	k		integer
	|	l		heap location
	|	el		array element
	|	$\textbf{Many}\, e$		!-introduction
	|	$\textbf{let}\, \textbf{Many}\, x = e\, \textbf{in}\, e'$		!-elimination
	|	$\textbf{fun}\, fc \to e$		frac. cap. abstraction
	|	$e[f]$		frac. cap. specialisation
	|	(e, e')		pair introduction
	|	$\textbf{let}\, (a, b) = e\, \textbf{in}\, e'$	bind $a \cup b$ in e'	pair elimination
	|	$\textbf{fun}\, x : t \to e$	bind x in e	abstraction
	|	$e\ e'$		application
	|	$\textbf{fix}\, (g, x : t, e : t')$	bind $g \cup x$ in e	fixpoint
	|	(e)	S	parentheses
C	::=			evaluation contexts
	|	$\textbf{let}\, x = [-]\, \textbf{in}\, e$	bind x in e	let binding
	|	$\textbf{let}\, () = [-]\, \textbf{in}\, e$		unit elimination
	|	$\textbf{if}\, [-]\, \textbf{then}\, e_1\, \textbf{else}\, e_2$		if
	|	$\textbf{Many}\, [-]$		!-introduction
	|	$\textbf{let}\, \textbf{Many}\, x = [-]\, \textbf{in}\, e$		!-elimination
	|	$\textbf{fun}\, fc \to [-]$		frac. cap. abstraction
	|	$[-][f]$		frac. cap. specialisation
	|	$([-], e)$		pair introduction
	|	$(v, [-])$		pair introduction
	|	$\textbf{let}\, (a, b) = [-]\, \textbf{in}\, e$	bind $a \cup b$ in e	pair elimination
	|	$[-]e$		application
	|	$v[-]$		application
Θ	::=			fractional capability environment
	|	$.$		
	|	Θ, fc		
Γ	::=			linear types environment
	|	$.$		
	|	$\Gamma, x : t$		
	|	Γ, Γ'		
Δ	::=			intuitionistic types environment
	|	$.$		
	|	$\Delta, x : t$		
σ	::=			heap
	|	$\{\}$		empty heap
	|	$\sigma \uplus \{l \mapsto_f m_{k_1,k_2}\}$		location l points to matrix m
$StepsTo$	::=			result of small step

$\mid$	$\langle \sigma, e \rangle$	heap and expression
$\mid$	**err**	error

$\boxed{\Theta \vdash f \ \mathsf{Cap}}$ Valid fractional capabilities

$$\frac{fc \in \Theta}{\Theta \vdash fc \ \mathsf{Cap}} \quad \textsc{WF\_Cap\_Var}$$

$$\frac{}{\Theta \vdash 1 \ \mathsf{Cap}} \quad \textsc{WF\_Cap\_Zero}$$

$$\frac{\Theta \vdash f \ \mathsf{Cap}}{\Theta \vdash \frac{1}{2} \cdot f \ \mathsf{Cap}} \quad \textsc{WF\_Cap\_Succ}$$

$\boxed{\Theta \vdash t \ \mathsf{Type}}$ Valid types

$$\frac{}{\Theta \vdash \mathbf{unit} \ \mathsf{Type}} \quad \textsc{WF\_Type\_Unit}$$

$$\frac{}{\Theta \vdash \mathbf{bool} \ \mathsf{Type}} \quad \textsc{WF\_Type\_Bool}$$

$$\frac{}{\Theta \vdash \mathbf{int} \ \mathsf{Type}} \quad \textsc{WF\_Type\_Int}$$

$$\frac{}{\Theta \vdash \mathbf{elt} \ \mathsf{Type}} \quad \textsc{WF\_Type\_Elt}$$

$$\frac{\Theta \vdash f \ \mathsf{Cap}}{\Theta \vdash f \ \mathbf{arr} \ \mathsf{Type}} \quad \textsc{WF\_Type\_Array}$$

$$\frac{\Theta \vdash t \ \mathsf{Type}}{\Theta \vdash !t \ \mathsf{Type}} \quad \textsc{WF\_Type\_Bang}$$

$$\frac{\Theta, fc \vdash t \ \mathsf{Type}}{\Theta \vdash \forall fc.t \ \mathsf{Type}} \quad \textsc{WF\_Type\_Gen}$$

$$\frac{\begin{array}{c}\Theta \vdash t \ \mathsf{Type} \\ \Theta \vdash t' \ \mathsf{Type}\end{array}}{\Theta \vdash t \otimes t' \ \mathsf{Type}} \quad \textsc{WF\_Type\_Pair}$$

$$\frac{\begin{array}{c}\Theta \vdash t \ \mathsf{Type} \\ \Theta \vdash t' \ \mathsf{Type}\end{array}}{\Theta \vdash t \multimap t' \ \mathsf{Type}} \quad \textsc{WF\_Type\_Lolly}$$

$\boxed{\Theta; \Delta; \Gamma \vdash e : t}$ Typing rules for expressions

$$\frac{}{\Theta; \Delta; \cdot, x : t \vdash x : t} \quad \textsc{Ty\_Var\_Lin}$$

$$\frac{x : t \in \Delta}{\Theta; \Delta; \cdot \vdash x : t} \quad \textsc{Ty\_Var}$$

$$\frac{\begin{array}{c}\Theta; \Delta; \Gamma \vdash e : t \\ \Theta; \Delta; \Gamma', x : t \vdash e' : t'\end{array}}{\Theta; \Delta; \Gamma, \Gamma' \vdash \mathbf{let}\ x = e \ \mathbf{in}\ e' : t'} \quad \textsc{Ty\_Let}$$

$$\frac{}{\Theta; \Delta; \cdot \vdash () : \mathbf{unit}} \quad \textsc{Ty\_Unit\_Intro}$$

$$\frac{\begin{array}{c}\Theta; \Delta; \cdot \vdash e : \mathbf{unit} \\ \Theta; \Delta; \Gamma \vdash e' : t\end{array}}{\Theta; \Delta; \Gamma \vdash \mathbf{let}\ () = e \ \mathbf{in}\ e' : t} \quad \textsc{Ty\_Unit\_Elim}$$

$$\frac{}{\Theta; \Delta; \cdot \vdash \mathbf{true} : \mathbf{bool}} \quad \textsc{Ty\_Bool\_True}$$

$$\frac{}{\Theta; \Delta; \cdot \vdash \mathbf{false} : \mathbf{bool}} \quad \textsc{Ty\_Bool\_False}$$

$$\frac{\begin{array}{l}\Theta; \Delta; \Gamma \vdash e : !\mathbf{bool}\\ \Theta; \Delta; \Gamma' \vdash e_1 : t'\\ \Theta; \Delta; \Gamma' \vdash e_2 : t'\end{array}}{\Theta; \Delta; \Gamma, \Gamma' \vdash \mathbf{if}\ e\ \mathbf{then}\ e_1\ \mathbf{else}\ e_2 : t} \quad \textsc{Ty\_Bool\_Elim}$$

$$\frac{}{\Theta; \Delta; \cdot \vdash k : \mathbf{int}} \quad \textsc{Ty\_Int\_Intro}$$

$$\frac{}{\Theta; \Delta; \cdot \vdash el : \mathbf{elt}} \quad \textsc{Ty\_Elt\_Intro}$$

$$\frac{\begin{array}{l}\Theta; \Delta; \cdot \vdash v : t\\ v \neq l\end{array}}{\Theta; \Delta; \cdot \vdash \mathbf{Many}\ v : !t} \quad \textsc{Ty\_Bang\_Intro}$$

$$\frac{\begin{array}{l}\Theta; \Delta; \Gamma \vdash e : !t\\ \Theta; \Delta, x : t; \Gamma' \vdash e' : t'\end{array}}{\Theta; \Delta; \Gamma, \Gamma' \vdash \mathbf{let}\ \mathbf{Many}\ x = e\ \mathbf{in}\ e' : t'} \quad \textsc{Ty\_Bang\_Elim}$$

$$\frac{\begin{array}{l}\Theta; \Delta; \Gamma \vdash e : t\\ \Theta; \Delta; \Gamma' \vdash e' : t'\end{array}}{\Theta; \Delta; \Gamma, \Gamma' \vdash (e, e') : t \otimes t'} \quad \textsc{Ty\_Pair\_Intro}$$

$$\frac{\begin{array}{l}\Theta; \Delta; \Gamma \vdash e_{12} : t_1 \otimes t_2\\ \Theta; \Delta; \Gamma', a : t_1, b : t_2 \vdash e : t\end{array}}{\Theta; \Delta; \Gamma, \Gamma' \vdash \mathbf{let}\ (a, b) = e_{12}\ \mathbf{in}\ e : t} \quad \textsc{Ty\_Pair\_Elim}$$

$$\frac{\begin{array}{l}\Theta \vdash t'\ \mathsf{Type}\\ \Theta; \Delta; \Gamma, x : t' \vdash e : t\end{array}}{\Theta; \Delta; \Gamma \vdash \mathbf{fun}\ x : t' \to e : t' \multimap t} \quad \textsc{Ty\_Lambda}$$

$$\frac{\begin{array}{l}\Theta; \Delta; \Gamma \vdash e : t' \multimap t\\ \Theta; \Delta; \Gamma' \vdash e' : t'\end{array}}{\Theta; \Delta; \Gamma, \Gamma' \vdash e\ e' : t} \quad \textsc{Ty\_App}$$

$$\frac{\Theta, fc; \Delta; \Gamma \vdash e : t}{\Theta; \Delta; \Gamma \vdash \mathbf{fun}\ fc \to e : \forall fc.t} \quad \textsc{Ty\_Gen}$$

$$\frac{\begin{array}{l}\Theta \vdash f\ \mathsf{Cap}\\ \Theta; \Delta; \Gamma \vdash e : \forall fc.t\end{array}}{\Theta; \Delta; \Gamma \vdash e[f] : t[f/fc]} \quad \textsc{Ty\_Spc}$$

$$\frac{\Theta; \Delta, g : t \multimap t'; \cdot, x : t \vdash e : t'}{\Theta; \Delta; \cdot \vdash \mathbf{fix}\ (g, x : t, e : t') : !(t \multimap t')} \quad \textsc{Ty\_Fix}$$

$\boxed{\langle \sigma, e \rangle \to \mathit{StepsTo}}$ operational semantics

$$\frac{}{\langle \sigma, \mathbf{let}\ () = ()\ \mathbf{in}\ e \rangle \to \langle \sigma, e \rangle} \quad \textsc{Op\_Let\_Unit}$$

$$\frac{}{\langle \sigma, \mathbf{let}\ x = v\ \mathbf{in}\ e \rangle \to \langle \sigma, e[x/v] \rangle} \quad \textsc{Op\_Let\_Var}$$

$$\frac{}{\langle \sigma, \mathbf{if}\ (\mathbf{Many}\ \mathbf{true})\ \mathbf{then}\ e_1\ \mathbf{else}\ e_2 \rangle \to \langle \sigma, e_1 \rangle} \quad \textsc{Op\_If\_True}$$

$$\frac{}{\langle \sigma, \mathbf{if}\ (\mathbf{Many}\ \mathbf{false})\ \mathbf{then}\ e_1\ \mathbf{else}\ e_2 \rangle \to \langle \sigma, e_2 \rangle} \quad \textsc{Op\_If\_False}$$

$$\frac{}{\langle \sigma, \textbf{let}\,\textbf{Many}\; x = \textbf{Many}\; v\; \textbf{in}\; e\rangle \to \langle \sigma, e[x/v]\rangle}\quad \textsc{Op\_Let\_Many}$$

$$\frac{e_1 = e[g/\textbf{let}\,\textbf{Many}\; g = \textbf{fix}\,(g, x : t, e : t')\,\textbf{in}\,\textbf{fun}\; x : t \to e]}{\langle \sigma, \textbf{let}\,\textbf{Many}\; g = \textbf{fix}\,(g, x : t, e : t')\,\textbf{in}\; e'\rangle \to \langle \sigma, e'[g/\textbf{fun}\; x : t \to e_1]\rangle}\quad \textsc{Op\_Let\_Fix}$$

$$\frac{}{\langle \sigma, (\textbf{fun}\, fc \to v)[f]\rangle \to \langle \sigma, v[fc/f]\rangle}\quad \textsc{Op\_Frac\_Cap}$$

$$\frac{}{\langle \sigma, (\textbf{fun}\; x : t \to e)\; v\rangle \to \langle \sigma, e[x/v]\rangle}\quad \textsc{Op\_App}$$

$$\frac{\langle \sigma, e\rangle \to \langle \sigma', e'\rangle}{\langle \sigma, C[e]\rangle \to \langle \sigma, C[e']\rangle}\quad \textsc{Op\_Context}$$

$$\frac{0 \le k_1, k_2}{\langle \sigma, \textbf{matrix}\; k_1\; k_2\rangle \to \langle \sigma \uplus \{l \mapsto_1 M_{k_1,k_2}\}, l\rangle}\quad \textsc{Op\_Matrix}$$

$$\frac{}{\langle \sigma \uplus \{l \mapsto_1 m_{k_1,k_2}\}, \textbf{free}\; l\rangle \to \langle \sigma, ()\rangle}\quad \textsc{Op\_Free}$$

$$\frac{}{\langle \sigma \uplus \{l \mapsto_f m_{k_1,k_2}\}, \textbf{share}\; l\rangle \to \langle \sigma \uplus \{l \mapsto_{\frac{1}{2}\cdot f} m_{k_1,k_2}\} \uplus \{l \mapsto_{\frac{1}{2}\cdot f} m_{k_1,k_2}\}, (l, l)\rangle}\quad \textsc{Op\_Share}$$

$$\frac{f \le 1}{\langle \sigma \uplus \{l \mapsto_{\frac{1}{2}\cdot f} m_{k_1,k_2}\} \uplus \{l \mapsto_{\frac{1}{2}\cdot f} m_{k_1,k_2}\}, \textbf{unshare}\; l\; l\rangle \to \langle \sigma \uplus \{l \mapsto_f m_{k_1,k_2}\}, l\rangle}\quad \textsc{Op\_Unshare\_Eq}$$

$$\frac{l \ne l'}{\langle \sigma \uplus \{l \mapsto_{\frac{1}{2}\cdot f} m_{k_1,k_2}\} \uplus \{l' \mapsto_{\frac{1}{2}\cdot f} m_{k_1,k_2}\}, \textbf{unshare}\; l\; l'\rangle \to \textbf{err}}\quad \textsc{Op\_Unshare\_Neq}$$

$$\frac{\sigma' = \sigma \uplus \{l_1 \mapsto_{fc_1} m_{1\,k_1,k_2}\} \uplus \{l_2 \mapsto_{fc_2} m_{2\,k_2,k_3}\}}{\langle \sigma' \uplus \{l_3 \mapsto_1 m_{1\,k_1,k_3}\}, \textbf{gemm}\; l_1\; l_2\; l_3\rangle \to \langle \sigma' \uplus \{l_3 \mapsto_1 (m_1\, m_2 + m_3)_{k_1,k_3}\}, ((l_1, l_2), l_3)\rangle}\quad \textsc{Op\_Gemm\_Match}$$

$$\frac{\begin{array}{l} k_2 \ne k_2' \\ \sigma' = \sigma \uplus \{l_1 \mapsto_{fc_1} m_{1\,k_1,k_2}\} \uplus \{l_2 \mapsto_{fc_2} m_{2\,k_2',k_3}\}\end{array}}{\langle \sigma' \uplus \{l_3 \mapsto_1 m_{1\,k_1,k_3}\}, \textbf{gemm}\; l_1\; l_2\; l_3\rangle \to \textbf{err}}\quad \textsc{Op\_Gemm\_Mismatch}$$

$$
\begin{aligned}
\mathcal{V}_k[\![\mathbf{unit}]\!] &= \{(\{\}, *)\} \\[2ex]
\mathcal{V}_k[\![\mathbf{bool}]\!] &= \{(\{\}, true), (\{\}, false)\} \\[2ex]
\mathcal{V}_k[\![\mathbf{int}]\!] &= \{(\{\}, n) \mid 2^{-63} \le n \le 2^{63} - 1\} \\[2ex]
\mathcal{V}_k[\![\mathbf{elt}]\!] &= \{(\{\}, f) \mid f \text{ a IEEE Float64 }\} \\[2ex]
\mathcal{V}_k[\![f\,\mathbf{mat}]\!] &= \{(\{l \mapsto_{2^{-f}} \_\}, l)\} \\[2ex]
\mathcal{V}_k[\![!(t' \multimap t'')]\!] &= \{(\{\}, \mathbf{Many}\, v) \mid (\{\}, v) \in \mathcal{V}_k[\![t' \multimap t'']\!]\} \\
&\quad \cup \{(\{\}, \mathbf{fix}(g, x : t, e : t')) \mid \forall\, j \le k, (\sigma', v') \in \mathcal{V}_j[\![t']\!]. \\
&\qquad\qquad (\sigma', \mathbf{let}\,\mathbf{Many}\; g = \mathbf{fix}\,(g, x : t, e : t')\;\mathbf{in}\; g\, v) \in \mathcal{C}_j[\![t']\!]\} \\[2ex]
\mathcal{V}_k[\![!t]\!] &= \{(\{\}, \mathbf{Many}\, v) \mid \neg(\exists t', t''.\; t = t' \multimap t'') \wedge (\{\}, v) \in \mathcal{V}_k[\![t]\!]\} \\[2ex]
\mathcal{V}_k[\![\forall fc.\; t]\!] &= \{(\sigma, \forall fc.\; v) \mid \forall f.\; (\sigma, v[fc/f]) \in \mathcal{V}_k[\![t[fc/f]]\!]\} \\[2ex]
\mathcal{V}_k[\![t' \otimes t'']\!] &= \{(\sigma, \langle v', v'' \rangle) \mid \exists \sigma', \sigma''.\; (\sigma', v') \in \mathcal{V}_k[\![t]\!] \wedge (\sigma'', v'') \in \mathcal{V}_k[\![t'']\!] \wedge \sigma = \sigma' \star \sigma''\} \\[2ex]
\mathcal{V}_k[\![t \multimap t']\!] &= \{(\sigma, \mathbf{fun}\, x : t \to e) \mid \forall j \le k, (\sigma', v') \in \mathcal{V}_j[\![t']\!].\; \sigma \star \sigma' \text{ defined } \Rightarrow (\sigma \star \sigma', (\mathbf{fun}\, x : t \to e) v') \in \mathcal{C}_j[\![t \\[2ex]
\mathcal{C}_k[\![t]\!] &= \{(\sigma_s, e) \mid \forall\, j < k, \sigma_r.\; \sigma_s \star \sigma_r \text{ defined } \Rightarrow \\
&\qquad \langle \sigma_s \star \sigma_r, e \rangle \to^j \mathbf{err} \vee \exists \sigma_f, v.\; \langle \sigma_s \star \sigma_r, e \rangle \to^j \langle \sigma_f \star \sigma_r, v \rangle \in \mathcal{V}_{k-j}[\![t]\!]\} \\[2ex]
\mathcal{I}_k[\![\cdot]\!]\theta &= [] \\[2ex]
\mathcal{I}_k[\![\Delta, x : t]\!]\theta &= \{\delta[x \mapsto v_x] \mid \delta \in \mathcal{I}_k[\![\Delta]\!]\theta \wedge (\{\}, v_x) \in \mathcal{V}_k[\![\theta(t)]\!]\} \\[2ex]
\mathcal{L}_k[\![\cdot]\!]\theta &= \{(\{\}, [])\} \\[2ex]
\mathcal{L}_k[\![\Gamma, x : t]\!]\theta &= \{(\sigma \uplus \sigma_x, \gamma[x \mapsto v_x]) \mid (\sigma, \gamma) \in \mathcal{L}_k[\![\Gamma]\!]\theta \wedge (\sigma_x, v_x) \in \mathcal{V}_k[\![\theta(t)]\!]\} \\[2ex]
[\![\Theta; \Delta; \Gamma \vdash e : t]\!] &= \forall \theta, k, \delta, \gamma, \sigma.\; \mathrm{dom}(\Theta) = \mathrm{dom}(\theta) \wedge (\sigma, \delta) \in \mathcal{L}_k[\![\Gamma]\!]\theta \wedge \gamma \in \mathcal{I}_k[\![\Delta]\!]\theta \Rightarrow \\
&\qquad (\sigma, \gamma(\delta(e))) \in \mathcal{C}_k[\![\theta(t)]\!]
\end{aligned}
$$